IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BOARD OF PATENT APPEALS AND INTERFERENCES

Appl. No.	:	10/620,988
Appellant	:	Everett, Ron
Filed	:	July 16, 2003
Title	:	Data Base and Knowledge Operating System
	:	
Group Art Unit	:	2162
Examiner	:	Myint, Dennis Y.
	:	
Docket No.	:	49593-00003

SUPPLEMENTAL APPEAL BRIEF

Real Party in Interest

The real party in interest is the sole inventor, Mr. Ron Everett, a Canadian citizen residing in Montreal, Canada.

Related Appeals and Interferences

There are no related appeals and interferences.

Status of Claims

Claims 1-3, 5, 7-9, 11, 13, 15-37, 40-62 and 82-96 are currently pending in the application. Claims 4, 6, 10, 12, 14, 38-39 have been cancelled. Claims 63-81 have been withdrawn.

Of the currently pending claims, Claims 1-3, 5, 7-9, 11, 13, 15-37, 40-62 and 81-96 have been rejected and claims 25, 26, 35, 37, 41, 43, 45, 46 and 91 have been objected to.

All currently pending claims are the subject of this appeal. The claims on appeal are reproduced in the Claims Appendix.

Status of Amendments

An amendment after final was filed on August 3, 2007, accompanied by a Request for Continued Examination in which claims 1, 9, 11, 13, 15, 16, 21, 33, 40, 42, 44, 47, 52, 53, 82 and 85 were amended, and claim 2 was newly added. The examiner has indicated in the office action of September 12, 2007 that these amendments have been entered. The claims listed in the Claim s Appendix are therefore the claims as amended in the 08/03/2007 Response and Amendment.

Summary of the Claimed Subject Matter

Independent Claim 1 describes a data management system wherein each data instance is stored in a separate, independent data structure (*See* Specification, ¶¶ 180, 219-226, Figs. 12, 13). The data structures encapsulate within themselves references to other data structures which hold associated data instances (*See* Specification, ¶¶181, 209, Fig. 14). The data structures can be manipulated to add, delete and search specific data items (*See* Specification, ¶¶194, 272-277, Fig. 10).

Independent Claim 52 describes a data management system wherein each data instance stored in a separate, independent data structure, is referenced by a logical index (*See* Specification, ¶¶ 207-218, Fig. 14). The logical index not only uniquely identifies the data

instance, but also encodes its location on a physical media (*See* Specification ¶¶ 207, 278-287, Fig. 16).

Independent Claim 82 describes a method of converting a conventional database to a database having the structure as described in the application. First, data structures containing data instances are created. Next, associations between the data instances are created and stored with each data structure (*See* Specification, ¶¶ 308-331, Figs. 2-7).

Independent Claim 85 is similar to Claim 1, except that it requires the encapsulated references to associated data structures to be mutual. That is, if a first data structure encapsulates a pointer to a second data structure, indicating an association between their encapsulated data instances, the second data structure must also encapsulate a pointer to the first data structure (*See* Specification, ¶¶ 181, 197-201, Figs. 5, 14).

Grounds of Rejection to be Reviewed on Appeal

1. Has the examiner mis-characterized the subject matter of claims 1-3, 5, 7-9, 11, 13, 15-37, 40-62 and 81-96 as non-functional descriptive material and applied the incorrect standard for patentable subject matter?

2. Are claims 1, 3, 7, 9, 11, 13, 15, 16, 53, 82-87, 89 and 92 patentable in view of various combinations of U.S. Patent 6,609,132 (White), U.S. Published Application 2005/0044079 (Abineri) and U.S. Published Application 2003/0110513 (Plourde).

Argument

I. Claims 1-3, 5, 7-9, 11, 13, 15-37, 40-62 and 81-96 contain subject matter that is functional descriptive material, and is therefore statutory when recorded on a computer-readable medium.

The examiner has rejected independent claims 1, 52, 82 and 85, as well as their dependent claims, stating that they represent non-functional descriptive material, and, as such, are non-statutory under § 101, even when recorded on a computer-readable medium.

The Appellant submits that the examiner has mis-characterized this material as *non-functional*, descriptive material, and that the subject matter of these claims should be properly classified *functional*, descriptive material, that becomes statutory when recorded on a computer-readable medium.

MPEP § 2106.01 clearly states as follows:

> Descriptive material can be characterized as either "functional descriptive material" or "nonfunctional descriptive material." In this context, **"functional descriptive material" consists of data structures and computer programs which impart functionality when employed as a computer component. (The definition of "data structure" is "a physical or logical relationship among data elements, designed to support specific data manipulation functions."** [ref deleted] "Nonfunctional descriptive material" includes but is not limited to music, literary works, and a compilation or mere arrangement of data.
>
> Both types of "descriptive material" are nonstatutory when claimed as descriptive material *per se* [ref. deleted]. **When functional descriptive material is recorded on some computer-readable medium, it becomes structurally and functionally interrelated to the medium and will be statutory** in most cases since use of technology permits the function of the descriptive material to be realized.

And, specifically regarding data structures, under heading 1, the MPEP clearly states:

Data structures not claimed as embodied in computer-readable media are descriptive material *per se* and are not statutory because they are not capable of causing functional change in the computer. [ref. deleted] Such claimed data structures do not define any structural and functional interrelationships between the data structure and other claimed aspects of the invention which permit the data structure's functionality to be realized. In contrast, **a claimed computer-readable medium encoded with a data structure defines structural and functional interrelationships between the data structure and the computer software and hardware components which permit the data structure's functionality to be realized, and is thus statutory.**

The MPEP is clear in stating that data structures which impart functionality are to be considered functional descriptive material ("functional descriptive material" consists of data structures ... "which impart functionality when employed as a computer component"). The claimed subject matter meets the definition of a data structure provided in the MPEP (a physical or logical relationship among data elements, designed to support specific data manipulation functions). Although the Appellant does not believe it is necessary, limitations have been added to the independent claims which recite functionality and data manipulation capability ("...data instances can be added, removed and searched"). Therefore, the subject matter of the claims must be recognized as being functionally descriptive and therefore statutory when recorded on a computer-readable medium.

The examiner has repeatedly and incorrectly classified the claimed subject matter as being non-functional descriptive material, and has cited sections of the MPEP dealing with non-functional descriptive material in his rejection of the claims under § 101.

Further, the Appellant has modified the claims as they appear in the Claims Appendix to include not only the functional limitations noted above, but also limitations that the data structures be stored on a computer-readable medium. The MPEP clearly states the following: "a

claimed computer-readable medium encoded with a data structure defines structural and functional interrelationships between the data structure and the computer software and hardware components which permit the data structure's functionality to be realized, **and is thus statutory**." As such, the Appellant submits that the subject matter, as claimed, represents functional, descriptive subject matter which is statutory based upon is storage on a computer-readable medium, and that the examiner has erred in his application of sections of the MPEP dealing specifically with non-functional descriptive material.

II. Claims 1, 3, 7, 9, 11, 13, 15, 16, 53, 82-87, 89 and 92 are patentable in view of various combinations of U.S. Patent 6,609,132 (White), U.S. Published Application 2005/0044079 (Abineri) and other cited art

a. White teaches away from the present invention.

The Examiner has rejected claims 1, 3, 7, 9, 11, 13, 15, 16, 53, 82-87, 89 and 92 under 35 U.S.C. § 103(a) as being unpatentable over White in view of Abineri. The Appellant respectfully submits that the application of White is not proper in this instance because White teaches away from the invention in that White requires that its data items and the information relating to the relationships between data items be stored in separate places, typically in separate relational database tables. White states as follows:

> Preferably, the data structure (i.e., data records) representing a given relation linking two or more objects **are separate from (and**

indirectly coupled to) the data structures representing these two or more objects. (emphasis added)

See White, column 5, lines 22-26.

This is also clearly evident in Figures 2 and 3 of White, which shows this information being stored in separate relational database tables, and in various other places in the White specification, in which a description of the use of tables in White is explained.

White states the following:

> Figs. 2 and 3 illustrate an exemplary embodiment of logical data structures representing the inventive object data model of the present invention, including a plurality of objects (Object A, Object B, Object C, Object D as shown) each having a plurality of attributes (as data members) **for storing information that describes characteristics of the corresponding object. The attributes of a given object may be used to encapsulate data and/or link to software functionality and/or processes pertinent to the given object.** As shown in Fig. 3, a Type Table Entry for a given object type includes one or more object identifiers (or pointers or keys) that identify the objects that belong to the given object type.
>
> Fig. 2 illustrates **the logical data structures representing an object relation** (which stores textual annotation characterizing the semantics of a relationship linking two (or more) objects as described above), including a Relation Table Entry, Relation Object Table Entry and Modifier Table Entry. (emphasis added)

See White, column 6, line 66 through column 7, line 17.

As shown in the above passages from White, only characteristics of a given data object or "software functionality and/or processes pertinent to the given object" are stored as embedded objects in the White system. No disclosure is made in White of the storage of relationship information as embedded objects in the same storage structure in which the data item itself is stored. As shown in Figure 2 of White, and as referenced in the passage above, there are separate logical data structures for representing the relationship information between objects.

The Appellant therefore respectfully submits that White clearly discloses a system wherein the relationship data, which specifies the relationships between data objects, be stored separate from the data objects themselves. This teaches in direct opposition to the present invention, in which the relationship information is encapsulated within the data structure in which the data objects themselves are stored, as evidenced by the limitation stating "...each of said data structures also encapsulating references to other of said independent data structures encapsulating associated data instances...".

As a result of this teaching of White, the Appellant submits that the combination of White with any other reference is improper because White unequivocally teaches away from the present invention by requiring relationship information (both relationships & relationship types) to be stored separately from the data items.

The Appellant had also previously amended the independent claims of the application to recite the limitation that the data be stored in non-tabular form, which further clarifies the

distinction between White and the present invention, which specifically teaches away from the use of tables of any kind.

b. Abineri does not teach a database in "non-tabular" form

The examiner also stated that White does not explicitly teach the limitation "in non-tabular form." The Appellant respectfully submits that White not only does not teach this limitation but explicitly teaches away from this limitation. The Examiner further states that Abineri teaches this limitation in paragraphs 61-66. Abineri teaches a display generator for an object oriented database. Paragraphs 61-66 describes a tree structure consisting of objects having parent/child relationships with other objects. The Appellant directs the Board's attention to paragraph 35 of Abineri, which describes the tree structure and the objects and classes therein. Paragraph 40 describes the primary objective of Abineri, which is the conversion of a flat database file (i.e., a non-object oriented file) to an object orientated database for purposes of display in a tree structure. Abineri never actually discusses the structure of the original flat file database nor does he discuss the implementation of the object oriented database, which can be implemented in any manner well known in the art, including in a tabular form as described in White. Because there is no specific teaching of a database in non-tabular form in Abineri, the Appellant respectfully submits that this limitation of the claims is not taught by the combination of White and Abineri.

Even for the sake of argument, if Abineri did disclose a database in non-tabular form, it makes no sense to combine the teaching of a database in non-tabular form with a database that is

implemented via tables, as is the White database. As a result, the Appellant respectfully submits that this combination does not teach what the Examiner states and further that the combination of the two references is improper due to a lack of teaching, suggestion or motivation within either of the references to make the combination.

c. The combination of White, Abineri and Plourde does not disclosed the claimed method of claims 82-84.

With respect to claims 82-87, the Appellant respectfully submits that neither White, Abineri or Plourde teaches a system in which all associations between a given data item and other data items in may be encapsulated within the same data structure in which the data item is stored.

Instead, the associations in Abineri happen through a tree structure in which each of the objects in the tree may have only one parent but multiple children. *See* Figures 3a and 3b for a pictorial representation of the tree structure. This limitation is not present in the present invention, which instead allows references to **all** associated data items to be encapsulated within each individual data structure containing a data item.

Claims 83 and 84 are dependent upon claim 82 and for the same reasons as with respect to claim 82, are not disclosed by the cited combination.

d. Rejections in view of other art

The Examiner has rejected claims 5, 8, 18-24, 31-34, 36, 47-48, 50-52, 54-60, 62, 88, 90 and 93 as being unpatentable over White in view of Abineri and further in view of Plourde and Koenke. The Appellant's previous comments with respect to White and Abineri apply here as well, that is, the combination of these two references is improper because White teaches away from the teachings of the present invention and, in fact, teaches away from Abineri in that it requires the use of tables, while the present application cannot work with tables. The Examiner states that the combination of White and Abineri fails to teach the limitation that the logical index is m dimensional and has n bits per dimension, but that Kroenke teaches this limitation. First, the Appellant respectfully submits that Kroenke, which teaches a computer system for allowing a user to create a relational database scheme, also teaches away from the present application. Kroenke refers to a model which uses a relational database table, while the present application specifically rejects that particular type of database in favor of the data instance centric model presented in the claims (i.e., each data instance contained in an independent data structure of common form). Further, the Examiner states that Kroenke teaches an object data model with indices which are m dimensional and have n bits per dimension. However this is not disclosed in the cited passage in Kroenke, which describes a subscript for an attribute consisting of a pair integers having the form "m.n", wherein "m" refers to the minimum cardinality and "n" refers to the maximum carnality of the attribute. The minimum cardinality is a minimum number of instances of the attribute in the data record and the maximum cardinality is a maximum number of instances of the attribute in the data record. The Examiner's interpretation of Kroenke appears to be incorrect in that it does not disclose an "m dimensional" index but discloses a one

dimensional attribute having a minimum number of instances and a maximum number of instances. This is not the same as the "m dimensional" index referred to in claim 5. Therefore, for this additional reason, the Appellant respectfully submits that the combination of White, Abineri and Kroenke does not disclose the claimed subject matter and further that Kroenke teaches away from the present application which makes its application to and combination with the other references improper as having no teaching, suggestion or motivation to do so.

With respect to claim 8, the Examiner states that White in view of Kroenke teaches the limitation of at least one dimension having a plurality of encapsulated references, citing White at column 7, lines 5-11 and column 7, lines 45-52. As previously stated, White in column 7 at lines 5-11 discloses a relational database table which has table entries defining the types of data objects and at column 7, lines 45-52 a relational database table having table entries for defining the types of relationships between data objects. As previously stated, the use of the relational database tables in White teach away from the present application, which encapsulates all relationship information within an independent data structure for each data object. As a result, these portions of White are not applicable to the present application. The Examiner further states that this concept is disclosed in Kroenke at column 6, lines 26-65. This portion of Kroenke discloses an object orientated system and describes, as is shown in Figure 2, the value attributes and group attributes and the types of the attributes which are found within a particular semantic representation of an object. Nothing in this portion of Kroenke or anywhere in Kroenke teaches a multi-dimensional index wherein each dimension of the index is related to a plurality of encapsulated references. Kroenke, in the cited passages, merely describes a typical object

oriented class structure wherein attributes are assigned to objects and inherited by instances of the objects.

With respect to claim 24, the Examiner states that White in view of Kroenke teaches the limitation that each of the dimensions of the logical index corresponds to a type of association. Neither White nor Kroenke disclose a database management system in which the grouping of encapsulated references corresponds to the type of association present between the grouped references and the encapsulating data instance.

With respect to claim 36, this claim contains the limitation that the encapsulated references are a logical index which uniquely identifies the data instance and also encodes the physical location of the associated data instances on the computer readable media. That is, the unique reference to each data instance not only provides a unique reference but also provides the physical location of the record on the computer readable media, whether it be a hard disk or a random access memory of the computer. Neither White nor Kroenke teach this limitation. The cited portions of White in columns 6 and 7 teach table entries having pointers to entries in other tables, such as might contain relationship information. No where in Kroenke or White is it disclosed that each data object has a unique reference nor does it disclose a unique reference also encoding a physical location on a computer readable media. This is simply not disclosed in any of the references cited.

With respect to claim 23, the Examiner states that White teaches the limitation that an identity in one or more of the m dimensions in the logical index indicates a member item in a container. Once again the Examiner refers back to the portion of table 7 which refers to data

object types within a relational table. White does not teach a multi-dimensional index and, therefore, cannot teach that an identity in one of the dimensions indicates membership in a container item. White, in fact, does not appear to discuss container items. The cited portion of White discloses a table containing object data types. This portion of White does not refer to membership in a container or, for that matter, any type of relation between any items in the database.

Claims 27-30 have been rejected under 35 U.S.C. § 103(a) as being unpatentable over White in view of Abineri and Kroenke and further in view of Walker. The Examiner refers to Walker to teach the limitations of Boolean mathematical operations, which the Appellant is willing to stipulate are well known in the prior art. However, within the context of the database system presented in the parent claims of claims 27-30, the use of Boolean mathematical operations to sort through lists of encapsulated data instances for purposes of satisfying search criteria is not disclosed in the cited references.

The Examiner has rejected claim 40 under 35 U.S.C. § 103(a) as being unpatentable over White in view of Abineri and further in view of Plourde and Bielak, stating that the combination of White and Abineri does not teach a plurality of encapsulated references representing ASCII characters, but that this is taught in Bielak. In the scope of the present application, encapsulated data instances representing ASCII characters provides for the inclusion of a single ASCII character as a single database item to which other database items are able to refer by encapsulation. This portion of Bielak merely teaches that data may be stored in ASCII format. The use of the term "ASCII" in this format does not indicate that the claimed limitations are

disclosed in that reference. Bielak does not come close to teaching that individual ASCII characters may be stored as individual data items in a database and be referred to by other data items which require the use of that particular ASCII character. The same argument applies for claim 42 regarding Unicode characters for which the Examiner has cited Eversol, and claim 44 wherein data instances representing the token members of the random token set of any data type for which the Examiner has cited Schwartz.

The Examiner has rejected claims 17, 49 and 61 under 25 U.S.C. § 103(a) as being unpatentable over White in view of Abineri and further in view of Plourde and Silberberg, et al. stating that Silberberg teaches the limitation of encapsulated references being a reference to data instances in another computing environment. Silberberg does teach distributed database accessing wherein other databases could be considered "other computing environments." However, Silberberg teaches a specific architecture for accessing the information stored in other computing environments which is different than the architecture described in the present application. The limitation of the parent claims to claim 17, 49 and 61 describe this architecture and are not disclosed by the combination of White and Abineri as discussed in detail above.

The Examiner has rejected claims 94-96 under 35 U.S.C. § 103(a) as being unpatentable over White, in view of Abineri and further in view of Plourde and Suver, stating that Suver discloses, at column 10, lines 9-27 the encapsulation of embedded elements. First, Suver teaches a system for embedding information in object-relational databases. The Appellant points out that the present application describes neither an object oriented nor a relational database but instead describes a "data instance centric" database. Object orientated databases describe databases in

which attributes are inherited from parent to child depending on the type of data being represented in the data object. Relational databases are databases set up using relational database tables. Therefore, the Appellant respectfully submits that there is no teaching, suggestion or motivation for citing Suver with respect to the present application. Nevertheless, Suver does not teach the embedding of objects within encapsulating data objects but includes the embedding of elements within records in a relational data base table. Therefore, the Applicant respectfully submits that Suver has been misinterpreted by the Examiner as disclosing embedding of objects within an encapsulated data instance.

Respectfully submitted,

Dennis M. Carleton
Registration No. 40,938
FOX ROTHSCHILD LLC
625 Liberty Avenue
Pittsburgh, PA 15222
(412) 394-5568

Attorney for Appellant

Claims Appendix

1. A data management system comprising:

    a. a plurality of independent data structures having a common form, each of said data structures encapsulating a single data instance;

    b. each of said data structures also encapsulating references to other of said independent data structures encapsulating associated data instances; and

    c. wherein said plurality of data structures are stored on a computer-readable media in non-tabular form and further wherein said data instances encapsulated in said data structures can be added, removed and searched.

2. The data management system of Claim 1 wherein each of said independent data structures also encapsulates a reference indicating the location of itself within a multi-dimensional organization of said data structures.

3. The data management system of claim 1 wherein a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to said first encapsulated data instance.

4. (Cancelled)

5. The data management system of claim 1 wherein:

a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to said first encapsulated data instance;

wherein each of said encapsulated references is a logical index which uniquely identifies each of said associated encapsulated data instances and also encodes the location of each of said associated encapsulated data instances on said computer-readable media; and

wherein said logical index is 'm' dimensional, and has 'n' bits per dimension.

6. (Cancelled)

7. The data management system of claim 1 wherein:

said encapsulated references are in at least one dimensions; and

each of said at least one dimensions corresponds to a type of association.

8. The data management system of claim 7 wherein each of said at least one dimensions has a plurality of said encapsulated references.

9. The data management system of claim 1 wherein said data structures are application independent and generally the same for all of said data instances.

10. (Cancelled)

11. The data management system of claim 3 wherein said data structures are application independent and generally the same for all of said data instances.

12. (Cancelled)

13. The data management system of claim 5 wherein said data structures are application independent and generally the same for all of said data instances.

14. (Cancelled)

15. The data management system of claim 7 wherein said data structures are application independent and generally the same for all of said data instances.

16. The data management system of claim 8 wherein said data structures are application independent and generally the same for all of said data instances.

17. The data management system of claim 1 wherein at least one of said encapsulated references is a reference to an encapsulated data instance in another computing environment.

18. The data management system of claim 1 wherein the encapsulated references of at least one of said encapsulated data instances are unique and the encapsulated references of at least two of said encapsulated data instances are generally identical.

19. The data management system of claim 1 including a plurality of pre-existing encapsulated data instances having established associations, wherein at least one new encapsulated data instance is associated with at least one of said preexisting encapsulated data instates.

20. The data management system of claim 1 including a plurality of pre-existing encapsulated data instances having established associations, wherein any of said pre-existing encapsulated data instances can be removed and disassociated from other pre-existing associated encapsulated data instances.

21. The data management system of claim 1 including a plurality of pre-existing encapsulated data instances having established associations, wherein new associations between at least two pre-existing encapsulated data instances can be added.

22. The data management system of claim 1 including a plurality of pre-existing encapsulated data instances having established associations, wherein any of said pre-existing associations between said pre-existing encapsulated data instances can be removed.

23. The data management system of claim 1 further comprising a search capability for finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing said known encapsulated data instances representing said selection criteria comprising the steps of:

accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

retrieving said specific unknown encapsulated data instances.

24. The data management system of claim 23 wherein said encapsulated references are embodied as logical indexes in a plurality of dimensions, each of said dimensions corresponding to a type of association, wherein said accessing further comprises accessing said encapsulated references from said dimensions specified in said selection criteria.

25. The data management system of claim 23 wherein said encapsulated references are 'm' dimensional logical indexes, each of which uniquely identifies and encodes the location of said associated encapsulated data instances on said computer readable media, wherein said encapsulated references are filtered by Boolean operations on at least one of said 'm' dimensional logical indexes.

26. The data management system of claim 24 wherein said encapsulated references are 'm' dimensional logical indexes, each of which uniquely identifies and encodes the location of said associated encapsulated data instances on said computer readable media, wherein said encapsulated references are filtered by Boolean operations on at least one of said 'm' dimensional logical indexes.

27. The data management system of claim 23 wherein said Boolean operations further comprise:

basic mathematical operators which result in the direct exclusion of at least one encapsulated reference from the result of said comparing in a single operation.

28. The data management system of claim 24 wherein said Boolean operations further comprise:

a basic mathematical operator which results in the direct exclusion of at least one encapsulated reference from the result of said comparing in a single operation.

29. The data management system of claim 25 wherein said Boolean operations further comprise:

a basic mathematical operator which results in the direct exclusion of at least one encapsulated reference from the result of said comparing in a single operation.

30. The data management system of claim 26 wherein said Boolean operations further comprise:

a basic mathematical operator which results in the direct exclusion of at least one encapsulated reference from the result of said comparing in a single operation.

31. The system of claim 1 wherein said encapsulated data instances have attributes of a user interface.

32. The system of claim 31 wherein said attributes of a user interface are selected from a group of user views, display elements, and data access methods.

33. The system of claim 1 further comprising searching said system wherein the encapsulated references of two or more different encapsulated data instances are used to derive desired results.

34. The system of claim 33 wherein said encapsulated references of two or more different encapsulated data instances are compared for at least one of commonality, similarity and difference to derive sets of references corresponding to said desired results.

35. The system of claim 34 wherein said encapsulated references of two or more different encapsulated data instances are stored in an order based on value and are compared for at

least one of commonality, similarity and difference to derive sets of references corresponding to said desired results.

36. The system of claim 33 wherein:

a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to said first encapsulated data instance;

wherein each of said encapsulated references is a logical index which uniquely identifies each of said associated encapsulated data instances and also encodes the location of each of said associated encapsulated data instances on said computer readable media; and

wherein said logical index is 'm' dimensional, and has 'n' bits per dimension; the encapsulated references of two or more different encapsulated data instances are compared for at least one of commonality, similarity and difference to derive sets of references corresponding to said desired results.

37. The system of claim 33 wherein:

each of said at least one dimensions has a plurality of said encapsulated references; and

said encapsulated references of two or more different encapsulated data instances are stored in an order based on value and are compared for at least one of commonality, similarity and difference to derive sets of references corresponding to said desired results.

38. (Cancelled)

39. (Cancelled)

40. The system of claim 1 further comprising:

a plurality of encapsulated data instances representing ASCII characters;

said data structures containing said encapsulated data instances representing ASCII characters also containing encapsulated references to encapsulated data instances using one or more of said corresponding ASCII characters; and

said data structures containing encapsulated data instances using one or more of said ASCII characters also containing encapsulated references to said encapsulated data instances representing said used ASCII characters.

41. The system of claim 40 wherein said encapsulated references with a given ASCII character data instance refer to other encapsulated data instances using said ASCII characters based on the position of said given ASCII character in the sequence of said ASCII characters in said encapsulated data instances.

42. The system of claim 1 further comprising:

a plurality of encapsulated data instances representing Unicode characters;

said data structures containing said encapsulated data instances representing Unicode characters also containing encapsulated references to encapsulated data instances using one or more said Unicode characters; and

said data structures encapsulated data instances using one or more of said Unicode characters also contains encapsulated references to said data instances representing said used Unicode characters.

43. The system of claim 42 wherein said encapsulated references with a given Unicode character data instance refer to other data instances using said Unicode characters based on the position of said given Unicode characters in the sequence of said Unicode characters in said encapsulated data instances.

44. The system of claim 1 further comprising:

a plurality of encapsulated data instances representing the tokens of a token set of any data type;

said data structures containing said data instances representing said tokens also containing encapsulated references to encapsulated data instances using one or more of said tokens; and

said data structures containing encapsulated data instances using one or more of said tokens also containing encapsulated references to said encapsulated data instances representing said used tokens.

45. The system of claim 44 wherein said encapsulated references for a given token data instance refer to other encapsulated data instances using said token based on the position of said given token in the sequence of said tokens in said encapsulated data instance.

46. The system claim 45 wherein:

said token set is selected from a group consisting of a set of graphic descriptors, a set of colors, a set of shapes, a set of glyphs, a set of waveforms, a set of frequency values, a set of audio frequency values, a defined set of symbols, and real numbers.

47. The data management system of claim 1 wherein:

said data structure is application independent and is generally the same for all of said data instances;

finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing known encapsulated data instances representing said selection criteria;

accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

retrieving said specific unknown encapsulated data instances.

48. The system of claim 47 further comprising searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

49. The data management system of claim 1 wherein:

at least one of said encapsulated references is a reference to an encapsulated data instance in another computing environment; and

a first data instance is encapsulated with references to associated data instances and each of said associated data instances are separately encapsulated with a reference to an encapsulated data instance.

50. The data management system of claim 1 wherein:

said encapsulated references of at least one of said encapsulated data instances is unique and said encapsulated references of at least two of said encapsulated data instance are generally identical;

finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing known encapsulated data instances representing said selection criteria;

accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

retrieving said specific unknown encapsulated data instances.

51. The data management system of claim 1 wherein:

said encapsulated references of at least one of said encapsulated data instances is unique and said encapsulated references of at least two of said encapsulated data instance are generally identical; and

searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

52. A data management system in a computing environment comprising:

a plurality of independent data structures having a common form, each encapsulating a single data instance;

wherein said plurality of data structures are stored on a computer-readable media in non-tabular form;

wherein each of said encapsulated references is a logical index which uniquely identifies each of said associated encapsulated data instances and also encodes the location of each of said associated encapsulated data instances on said computer-readable media; and

wherein said logical index is 'm' dimensional, and has 'n' bits per dimension, said encapsulated references are in at least one dimensions, and each of said at least one dimensions corresponds to a type of association.

53. The data management system of claim 52 wherein each of said data structures are application independent and generally the same for all of said data instances.

54. The data management system of claim 53 further comprising a search capability for finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing known encapsulated data instances representing said selection criteria comprising the steps of:

accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

retrieving said specific unknown encapsulated data instances.

55. The system of claim 54 wherein said encapsulated references of two or more different said encapsulated data instances are used to find said specific unknown encapsulated data instances.

56. The data management system of claim 55 wherein at least one of said encapsulated references is a reference to a encapsulated data instance in another computing environment.

57. The data management system of claim 56 wherein said encapsulated references of at least one of said encapsulated data instances is unique and said encapsulated references of at least two of said encapsulated data instances are generally identical.

58. The system of claim 57 further comprising searching said system wherein said encapsulated references of different said encapsulated data instances are used to derive desired results.

59. The data management system of claim 52 further comprising a search capability for finding specific unknown encapsulated data instances from a selection criteria of known encapsulated data instances by accessing known encapsulated data instances representing said selection criteria comprising the steps of:

accessing references encapsulated with said known encapsulated data instances representing said selection criteria;

using Boolean operations to compare said accessed encapsulated references to find references to said specific unknown encapsulated data instances; and

retrieving said specific unknown encapsulated data instances.

60. The system of claim 59 wherein said encapsulated references of two or more different said encapsulated data instances are used to find said specific unknown encapsulated data instances.

61. The data management system of claim 52 wherein at least one of said encapsulated references is a reference to a encapsulated data instance in another computing environment.

62. The data management system of claim 52 wherein said encapsulated references of at least one of said encapsulated data instances is unique and said encapsulated references of at least two of said encapsulated data instance are generally identical.

63-81. (Withdrawn)

82. A method to convert a non-data instance centric database to a data instance centric database comprising:

creating encapsulated data instances in said data instance centric database representing elements of said non-data-instance centric database schema and data elements of said non-data-instance centric database;

creating associations amongst the said data instances in said data instance centric database representing the relationships between said data elements and said schema elements of the non-data-instance centric database; and

storing said associations as a reference to each associated data instance stored within a independent data structure having a common form encapsulating the associated data instances, which are stored in non-tabular form on a computer-readable media.

83. The method of claim 82 wherein said converting is through a software agent which is a data instance in said data instance centric database.

84. The method of claim 82 wherein said non-data instance centric database includes a flat file.

85. A data management system comprising:

one or more items;

wherein each of said items encapsulates a data instance;

wherein items which are associated with each other encapsulate mutual references to each other;

wherein said items may be added, removed and searched; and

wherein said items are stored in non-tabular form on a computer-readable media.

86. The data management system of claim 85 wherein each of said items is represented in a fundamental data structure.

87. The data management system of claim 85 wherein each of said items has a unique reference associated therewith.

88. The data management system of claim 87 wherein said unique reference also serves as an index to physically locate said data instance associated with each of said items on said computer-readable media.

89. The data management system of claim 85 wherein said references to associated items are arranged in sets defining the type of association between said item and each of said other items referenced in said set.

90. The data management system of claim 87 wherein each of said references is an "m" dimensional index, each of said dimensions being "n" bits in length.

91. The data management system of claim 90 wherein "m" is 4 and "n" is 30.

92. The data management system of claim 85 wherein said items may act as containers for one or more other member items.

93. The data management system of claim 92 wherein membership of an item within a container item is indicated by an identity in one or more of said "m" dimensions in said logical index of said container item and each of said member items.

94. The data management system of claim 85 wherein each of said items may encapsulate embedded elements.

95. The data management system of claim 94 wherein said embedded elements are references to other items.

96. The data management system of claim 85 wherein said data instances may contain data of any type.

Evidence Appendix

No additional evidence is being submitted.

Related Proceedings Appendix

No related proceedings exist.